• Sun – MySQL Overview
• MySQL Performance Tuning
• Next Steps
About MySQL

- 13 Years of Development
- 400+ in Database Group
- 750+ Partners
- 70K+ Downloads Per Day

Customers across every major operating system, hardware vendor, geography, industry, and application type

High Performance ▪ Reliable ▪ Easy to Use
Serving Key Markets & Industry Leaders

Web / Web 2.0

OEM / ISV's

On Demand, SaaS, Hosting

Telecommunications

Enterprise 2.0

Open-source powers the Web
MySQL 5.4 Performance Tuning
Step by Step
MySql Server Performance Tuning Step-by-Step

Brian Miezejewski
Senior Manager
Professional Services
MySql Database group

bski@sun.com
Agenda

• Overview
• Step 1 - Storage Engines
• Step 2 - Connections
• Step 3 - Sessions
• Step 4 - Query Cache
• Step 5 - Queries
• Step 6 - Schema
• What if I need more help?
Overview

• Focus of this talk is the MySQL server, not:
  – Operating System, Disk performance, Network performance, etc.
• Cover the main steps
  – Show at least one example for each step
  – Examples are things I run into most commonly in the field
  – Include links to MySQL manual for additional information
• This will be technical!
• I can’t make you a performance tuning wizard in 45 minutes - PT Class is 4 day class
  – http://www.mysql.com/training/courses/performance_tuning.html
• MySQL Performance Forum
What you will need to know first

- The MySQL server is controlled by “System Variables”
  - `mysql> SHOW VARIABLES [LIKE <str>];`
  - `linux1> mysqladmin -u <user> -p variables`
  - Set Via:
    - `my.cnf`
    - `SET [GLOBAL] <variable>=<value>`
    - command line
- You monitor how well your system variables are configured using “Status Variables”
  - `mysql> SHOW STATUS [LIKE <str>];`
  - `linux1> mysqladmin -u <user> -p extended`
  - `linux1> mysqladmin ... ex -i 15 -r | grep ^v` 0 ``
- Enabling the slow query log
MySQL Enterprise Monitor w/Query Analyzer

- Single, consolidated view into entire MySQL environment
- Auto discovery of MySQL Servers, Replication Topologies
- *Problem Query Detection, Analysis and Tuning* – New!
- Customizable rules-based monitoring and alerts
- Identifies problems *before* they occur
- Reduces risk of downtime
- Makes it easier to scale-out without requiring more DBAs

"Virtual MySQL DBA" Assistant
Rules of Tuning

• Never make a change in production first
• Have a good benchmark or reliable load
• Start with a good baseline
• Only change 1 thing at a time
  – identify a set of possible changes
  – try each change separately
  – try in combinations of 2, then 3, etc.
• Monitor the results
  – Query performance - query analyzer, slow query log, etc.
    • throughput
    • single query time
    • average query time
  – CPU - top, vmstat
  – IO - iostat, top, vmstat, bonnie++
  – Network bandwidth
• Document and save the results
Were do I find a benchmark?

- Make your own
  - Can use general query log output
- DBT2
- mysqlslap MySQL 5.1 +
- Sysbench
- supersmack
  - [http://vegan.net/tony/supersmack/](http://vegan.net/tony/supersmack/)
- mybench
Step 1 MySQL Supports Multiple Storage Engines

- **MyISAM** - Original Storage Engine, great for web apps
- **InnoDB** - Robust transactional storage engine
- **Memory Engine** - Stores all data in **Memory**
- InfoBright - Large scale data warehouse with 10x or more compression
- Kickfire - Appliance based, Worlds fastest 100GB TPC-H

To see what tables are in what engines
  - `mysql> SHOW TABLE STATUS ;`

Selecting the storage engine to use is a **tuning decision**
  - `mysql> alter table tab engine=myisam ;`
Step 1

MyISAM

- Fastest storage engine 3x or more *when appropriate*
  - Most web applications
  - Perfect for web search databases
  - 80/20 read/modify or higher
  - pure inserts and deletes with partitions or merge engine
  - no transactions
  - reporting DB/ Data Warehouse

- Most compact data of all non-compressed engines
- Table locking
- Supports concurrent inserts
- Full-Text and Geospatial support
Step 1  MyISAM Tuning

- The primary tuning factor in MyISAM are its two caches:
  - `key_buffer_cache` - should be 25% of available memory
  - system cache - leave 75% of available memory free
- Available memory is:
  - All on a dedicated server, if the server has 8GB, use 2GB for the `key_buffer_cache` and leave the rest free for the system cache to use.
  - Percent of the part of the server allocated for MySQL, i.e. if you have a server with 8GB, but are using 4GB for other applications then use 1GB for the `key_buffer_cache` and leave the remaining 3GB free for the system cache to use.
- Maximum size for a single key buffer cache is 4GB
- You can define multiple key buffer’s
- For more details on configuring the MyISAM key cache see:
Step 1 Monitoring the MyISAM Key Buffer Cache

- `mysql>show status like 'Key%' ;`
- **Key_blocks_not_flushed** - Dirty key blocks not flushed to disk
- **Key_blocks_unused** - unused blocks in the cache
- **Key_blocks_used** - used Blocks in the cache
- **% of cache free**: `Key_blocks_unused / (Key_blocks_unused + Key_blocks_used)`
- **Key_read_requests** - key requests to the cache
- **Key_reads** - times a key read request went to disk
- **Cache read hit %**: `Key_reads / Key_read_requests`
- **Key_write_requests** - key write request to cache
- **Key_writes** - times a key write request went to disk
- **Cache write hit %**: `Key_writes / Key_write_request`
- `cat /proc/meminfo` to see the system cache in linux
  - MemFree + Cached = memory available for system cache
InnoDB

• Transactional and fully ACID compliant
• Behavior most like traditional databases such as Oracle, DB2, SQL Server, etc.
• Data size is normally 2-3 X MyISAM
• MVCC = Non-blocking reads in most cases
• Fast, reliable recovery from crashes with zero committed data loss
• **Always** clustered on the primary key
  – Lookups by primary key, very fast
  – Range scans on primary key also very fast
  – Non-Primary key lookups use the primary key to find the record, this means 2 key lookups
  – Important to keep primary key small
InnoDB

- Unlike MyISAM InnoDB uses a single cache for both index and data
  - `Innodb_buffer_pool_size` - should be 70-80% of available memory.
  - It is not uncommon for this to be very large, i.e. 44GB on a system with 40GB of memory
  - Make sure its not set so large as to cause swapping!
  - `mysql>show status like 'Innodb_buffer%';`
- InnoDB can use direct IO on systems that support it, Linux, FreeBSD, and Solaris.
  - `Innodb_flush_method = O_DIRECT`
- For more InnoDB tuning see
Step 2 Connections

- MySQL Caches the threads used by a connection
  - `thread_cache_size` - Number of threads to cache
  - Setting this to 100 or higher is not unusual
- Monitor `Threads_created` to see if this is an issue
  - Counts connections **not** using the thread cache
  - Should be less than 1-2 a minute
  - Usually only an issue if more than 1-2 a second
- Only an issue if you create and drop a lot of connections, i.e. PHP
- Overhead is usually about 250k per thread
- `Aborted_clients` -
- `Aborted_connections` -
Step 3 Sessions

• Some session variables control space allocated by each session (connection)
  – Setting these to small can give bad performance
  – Setting these too large can cause the server to swap!
  – Can be set by connection
    • `SET SORT_BUFFER_SIZE=1024*1024*128`
  – Set small by default, increase in connections that need it

• `sort_buffer_size` - Used for ORDER BY, GROUP BY, SELECT DISTINCT, UNION DISTINCT
  – Monitor `Sort_merge_passes` < 1-2 an hour optimal
  – Usually a problem in a reporting or data warehouse database

• Other important session variables
  – `read_rnd_buffer_size` - Set to 1/2 `sort_buffer_size`
  – `join_buffer_size` - (BAD) Watch `Select_full_join`
  – `read_buffer_size` - Used for full table scans, watch `Select_scan`
  – `tmp_table_size` - Max temp table size in memory, watch `Created_tmp_disk_tables`
Query Cache

- MySQL’s Jekyll and Hyde of performance tuning options, when it is useful it really helps, when it hurts, it really hurts
- MySQL Query Cache caches both the query and the full result set
  - `query_cache_type` - Controls behavior
    - 0 or OFF - Not used (buffer may still be allocated)
    - 1 or ON cache all unless `SELECT SQL_NO_CACHE (DEFAULT)`
    - 2 or DEMAND cache none unless `SELECT SQL_CACHE`
  - `query_cache_size` - Determines the size of the cache
- `mysql> show status like 'Qc%' ;`
- Gives great performance if:
  - Identical queries returning identical data are used often
  - No or rare inserts, updates or deletes
- Best Practice
  - Set to DEMAND
  - Add SQL_CACHE to appropriate queries
Step 5  Queries I

• Often the # 1 issue in overall performance
• **Always, Always have your slow query log on!**
  – Use: `log_queries_not_using_indexes`
  – Check it regularly
  – Use `mysqldumpslow`:
  – Best practice is to automate running `mysqldumpslow` every morning and email results to DBA, DBDev, etc.

• Understand and use **EXPLAIN**

• **Select_scan** - Number of full table scans
• **Select_full_join** - Joins without indexes
• MySQL Query Analyzer
The IN clause in MySQL is very fast!

- Select ... Where idx IN(1,23,345,456)
- Much faster than a join
- I have done tests with 80,000 items in the in list
  - 1,000-2,000 not unusual

Don’t wrap your indexes in expressions in Where

- Select ... Where func(idx) = 20 [index ignored]
  - Select .. Where idx = otherfunc(20) [may use index]
  - Best practice: Keep index alone on left side of condition

Avoid % at the start of LIKE on an index

- Select ... Where idx LIKE(‘ABC%’) can use index
- Select ... Where idx LIKE(‘%XYZ’) must do full table scan

Use union all when appropriate, default is union distinct!

Understand left/right joins and use only when needed

MySQL Query Analyzer – New!

- Centralized monitoring of Queries across all servers with no reliance on Slow Query Logs
- Aggregated view of query execution counts, time, and rows returned = total query expense
- Saves time/effort parsing atomic executions for total query expense

Saves time finding most expensive queries across all Production, Dev, and QA servers so SQL can be tuned.

“Finds code problems before your customers do.”
Step 6

Schema I

- Too many indexes slow down inserts/deletes
  - Use only the indexes you must have
  - Check often
- `mysql>show create table tabname ;`
- Don’t duplicate leading parts of compound keys
  - `index key123 (col1,col2,col3)`
  - `index key12 (col1,col2) <- Not needed!`
  - `index key1 (col1) <-- Not needed!`
- Use prefix indexes on large keys
- Best indexes are 16 bytes/chars or less
- Indexes bigger than 32 bytes/chars should be looked at very closely
  - should have there own cache if in MyISAM
- For large strings that need to be indexed, i.e. URLs, consider using a separate column using the MySQL MD5 to create a hash key.
• Size = performance, smaller is better
  – Size right! Do not automatically use 255 for VARCHAR
• Temp tables, most caches, expand to full size
• Use “procedure analyse” to determine the optimal types given the values in your table
  – mysql> select * from tab procedure analyse (64,2000) \G
• Consider the types:
• Compress large strings
  – Use the MySQL COMPRESS and UNCOMPRESS functions
  – Very important in InnoDB!
MySQL PS Performance Tuning

- **When:** Before go-live, or for application already deployed
- **What:** Tuning of database configuration, SQL, schema
  - Performance Tuning typically 3-5 days
  - Analysis with benchmark, load testing
  - Results typically 50-100% gain, sometimes 500%-1000%
- **Good for:**
  - Applications with poor response times, hit wall recently?
  - Solving system scalability issues
  - Reduction of hardware $$$ outlay
  - Prevention of new complex scaling architecture
  - Load testing, benchmarking, capacity planning

![Graph showing transactions/time, Starvation, Concurrency, and Response Time](image-url)
Case Study: Performance Tuning

• **Situation:**
  – Social Networking site having slow queries, crashes, & high disk i/o.
  – Large number of mysql servers being optimized

• **What we did:**
  – Benchmark based on general query log queries
  – The conversion of tables to InnoDB, and the optimization of these tables
  – Database configuration improvements and tuned for InnoDB
  – Over 67 schema changes including:
    • Adding multi-column indexes that are optimized for some queries
    • Column data type optimizations
    • The removal of unnecessary tables
    • The removal of unnecessary indexes

• **Results:**
  – 68% faster on warm benchmark
  – Dramatic reduction in disk i/o
  – No more database crashing and corruption
Case Study: Performance Tuning

Single Schema Benchmark

Benchmark Time (sec.)

<table>
<thead>
<tr>
<th>Benchmark Type</th>
<th>Cold</th>
<th>Warm</th>
</tr>
</thead>
<tbody>
<tr>
<td>MyISAM - Original</td>
<td>130</td>
<td>30</td>
</tr>
<tr>
<td>InnoDB with new indexes</td>
<td>60</td>
<td>25</td>
</tr>
</tbody>
</table>

Database system I/O wait %
Thank You
Next Steps

*MySQL 5.4 Benchmarks In-Depth – Allan Packer - Today!*


- Benchmarks
- Articles
- Archived Webinars
- Whitepapers
- Blogs
- Documentation
- Case Studies and More...

MySQL Consulting

- Architecture and Design
- Performance Tuning and Optimization
- High Availability
- Data Warehousing
- Remote DBA
- Migration
- Security
- Replication
- Back-up and Recovery

http://www.mysql.com/consulting/
MySQL Training

MySQL Performance Tuning (4 Days)

• Manage an increasing amount of data in your MySQL applications

• Monitor, diagnose problem areas and tune MySQL for optimal performance

• Write queries that take advantage of the MySQL 5.0 and 5.1 performance enhancements dealing with queries and indexing?

• Evaluate the application architecture for efficient design, structure, caching, number of connections and other factors affecting performance

www.mysql.com/training/courses/performance_tuning.html
MySQL Enterprise

Enterprise software & services delivered in an annual subscription

Database
- MySQL Enterprise Server
- Monthly Rapid Updates
- Quarterly Service Packs
- Hot Fix Program
- Indemnification

Monitoring
- Global Monitoring of All Servers
- Web-Based Central Console
- Built-in Advisors, Expert Advice
- Problem Query Detection/Analysis
- Specialized Scale-Out Help

Support
- Online Self help Knowledge Base
- 24 x 7 x 365 Problem Resolution
- Consultative Help
- High Availability and Scale Out
Sun’s Recommended Solutions for MySQL

MySQL’s Best Friend
Scale MySQL & Boost Performance with Sun Systems

See How Sun Optimizes MySQL
Watch Now

Learn How To:
» Scale MySQL for High Growth Web Apps
» Optimize MySQL Using Sun Systems
» Deploy MySQL for High Availability

Try Sun Systems for MySQL Free for 60 Days
Get MySQL 5.1

www.sun.com/mysqlsystems

X86 Performance ▪ Virtualization ▪ Backup
Multi-Tier Deployment ▪ Rich Media Storage
Questions?

Send them to
jimmy.guerrero@sun.com